HWZ Hochschule für Wirtschaft Zürich

Auftrag für Leistungsnachweis

Fach	Programming-Tools	**Bearbeitungszeit**	ca 7h
Titel	REST-Logger		
Studiengruppe	BWI-A20	**Anzahl Aufgaben**	Einzelarbeit
Abgabe-Datum	2.12.21 / 17:00	**Bewertung**	Geprüft wird der Code nach vorgegebenen von Funktionalen und Qualitäts-Kriterien. Anhand eines Fachgespräches wird überprüft, ob der Student den Code und die darunter liegenden theoretischen Grundlagen verstanden hat. (Gewicht: 100%)

Erlaubte Hilfsmittel	• Alles inkl. Google, PyCharm, … **ausser 1:1 Abschreiben oder Kopieren** ohne Deklaration als Kommentar im Code.
Auftrag	• Design und Implementation einer allgemeinen Logger Klasse • Entwickeln einer CLI Test-Applikation, welche Daten via einen REST-Aufruf von einem Web-Service abfragt (JSON respones) und nach verschiedenen Strategien in einem csv-File loggen kann.

Beschreibung

Daten in einem File zu loggen ist in verschiedensten Bereichen eine häufige Aufgabe. Sie entwickeln eine eigene Logger Klasse, welche die Informationen strukturiert in Form eines CSV-Files (Welches von Excel weiter verarbeitet werden könnte) abspeichert. Erstellen Sie ebenfalls im gleichen File eine CLI-Applikation, welche Sie zum Testen verwenden können.

Dabei soll das Log-File einen Header aus zwei Zeilen haben:

- Kommentar mit Filename und Start-Time, wann mit Loggen begonnen wurde (XML Syntax)
- Eine Titel-Zeile, welche die Spalten beschriftet

Die Logeinträge müssen mindestens folgende Elemente enthalten:

- Time-Stamp (Format ist eine Property der Klasse)
- Log-Level (DEBUG, INFO, WARNINGS, ERROR, CRITICAL)

Weiter müssen folgende Properties gesetzt werden können (über Setter und Initializer-Overloading):

- Delimiter (Default: „|“)
- File-Path und File-Name
- Anzahl Einträge, ab wann „gescrolled“ wird (alte Einträge gelöscht werden sollen)
- Aufzeichnungs-Strategie (Fixed Slices, OnlyChanges)
- Im Initializer muss gewählt werden können, ob neues File oder an bestehendes angehängt wird

Für die Applikation sind folgende Argumente (Properties) zu implementieren.

- Sample-Time
- URL für REST-Call
- Float und Integer Log-Daten werden ausserhalb der Log-Klasse

Functional and Quality-Requirements

- Lauffähige Test-Applikation rechtzeitig abgegeben
- Alles in einem File (ausnahmsweise für diese Aufgabenstellung)
- Filename: Vorname_Nachname_A19_DS.py (z.B. Rea_Vogel_A19_DS.py)
- Cleancode Regeln berücksichtigt
- Eigene Logger-Klasse vorhanden und Log-File enthält:
    - Header mit Filename, Creation Timestamp in XML-Syntax
    - Title mit Spalten beschriftet, CSV-Syntax
    - Pflichtfelder im Logfile Time-Stamp, Log-Level
    - Zwei verschiedenen Aufzeichnungs-Strategien implementiert
    - “Ringbuffer” implementiert
- Application
    - Eigene AppId gelöst
    - Timer verwendet
    - Sample-Time dynamisch
    - URL für REST-Call wird generiert (nicht Hard-Coded)
- Test
    - Test-Fälle dokumentiert
    - Test-Statistik vorhanden
    - Test-Abdeckung genügend
    - Test-Driven approach erkennbar

Detailbeschreibung / Examples

Beispiel eines Logfiles (im „OnlyChanges“ Mode)

```
# <Name>G:\LogFiles\2020\KM_2019_08_29.txt</Name>
Timestamp;Level;q [%] ;p_Low [bar];ShutOff-Valve;Water Pump
2019-08-29 18:07:01;INFO;0;2.34;CLOSED;OFF
2019-08-29 18:07:15;INFO;10;2.34;CLOSED;OFF
2019-08-29 18:07:16;INFO;10;2.3;OPEN;ON
2019-08-29 18:07:19;INFO;10;2.1;OPEN;ON
2019-08-29 18:07:19;INFO;20.5;2.1;OPEN;ON
2019-08-29 18:07:58;INFO;30.13;1.34;OFFEN;ON
2019-08-29 18:07:59;INFO;30.13;1.34;OFFEN;ON
2019-08-29 18:07:60;INFO;30.13;1.34;OFFEN;ON
2019-08-29 18:08:01;INFO;30.13;1.34;OFFEN;ON
```

Beispiel für REST-Call (OpenWeather)

OpenWeather Web-App: https://openweathermap.org/city/7287397

API doc and subscription: https://openweathermap.org/api

Get a free API-Key for Current weather data (Limited to 60 calls/minute or 1Mio calls/month ➔ alle 10” request = 267840)

```
# -------------------------------------------------------------------
# Name: WeatherLogger.py
#
# Description: Polling REST Service and write values to console
#
# Autor: Walter Rothlin
#
# History:
# 03-Dec-2020   Walter Rothlin      Initial Version ()
# 10-Oct-2021   Walter Rothlin      Adapted for BWI-A20
# -------------------------------------------------------------------

import requests
import json
import time

pollingTime = float(input("Polling-Time [s]:"))
serviceURL = "https://api.openweathermap.org/data/2.5/weather"
appId = "144747fd356c86e7926ca91ce78ce170"
while True:
    responseStr = requests.get(serviceURL + "?q=Uster&units=metric&lang=de&ap-
pid=" + appId)
    jsonResponse = json.loads(responseStr.text)

    temp = jsonResponse['main']['temp']
    pressure = jsonResponse['main']['pressure']
    humidity = jsonResponse['main']['humidity']
    lon = jsonResponse['coord']['lon']
    lat = jsonResponse['coord']['lat']
    cloud = jsonResponse['weather'][0]['description']

    print(temp, pressure, humidity, lon, lat, cloud)
    time.sleep(pollingTime)
```

JSON Response to
http://api.openweathermap.org/data/2.5/weather?q=Wangen%20SZ&appid=3836093dde650898eb014e6f27304646


```
▼ coord:
    lon:            8.89
    lat:            47.19
▼ weather:
  ▼ 0:
      id:           800
      main:         "Clear"
      description:  "clear sky"
      icon:         "01n"
  base:             "stations"
▼ main:
    temp:           275.35
    feels_like:     272.31
    temp_min:       274.82
    temp_max:       275.93
    pressure:       1022
    humidity:       85
  visibility:       10000
▼ wind:
    speed:          1.5
    deg:            151
▼ clouds:
    all:            0
  dt:               1606237231
▼ sys:
    type:           3
    id:             2006037
    country:        "CH"
    sunrise:        1606200100
    sunset:         1606232470
  timezone:         3600
  id:               2658054
  name:             "Wangen"
  cod:              200
```